RFC 5681 : TCP Congestion Control

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 11 septembre 2009

Date de publication du RFC : Septembre 2009

https://www.bortzmeyer.org/5681.html

Successeur du souvent cité RFC 2581 [1], ce RFC 5681 présente les algorithmes que doivent utiliser les mises en œuvres de TCP pour limiter les risques de congestion sur l'Internet. Comme TCP (RFC 793) est, de très loin, le principal protocole de transport de l'Internet, ces quatre algorithmes sont donc la principale ligne de défense contre la congestion.

Leurs noms en anglais sont tellement répandus qu'il n'est pas facile de concevoir une traduction. Essayons :

— Démarrage en douceur ("*slow start*"),
— Évitement de la congestion ("*congestion avoidance*"),
— Retransmission rapide ("*fast retransmit*"),
— Récupération rapide ("*fast recovery*").

Mis au point à l'origine par Van Jacobson, ces algorithmes ont d'abord été normalisés dans le RFC 1122 (avant les RFC 2001, une lecture très recommandée, et RFC 2581). Des RFC comme le RFC 3390 ont également contribué. Richard Stevens décrit ces algorithmes dans son livre "*TCP/IP Illustrated*" (explications dans le premier volume et code source dans le deuxième). Ce livre est de très loin le meilleur pour quiconque veut apprendre le délicat fonctionnement de TCP.

Si TCP envoyait des données à la vitesse maximale que permet la carte réseau et la machine, l'Internet s'écroulerait rapidement. Le but de ces algorithmes est donc de freiner TCP et la section 3 note qu'une mise en œuvre de TCP est autorisée à être plus prudente que ces algorithmes (à envoyer moins de données) mais pas à être plus violente.

La section 3.1 décrit le Démarrage en douceur. Le principe est, au démarrage d'une connexion TCP, de ne pas envoyer trop de données sans avoir vu qu'elles pouvaient arriver au même rythme. L'envoyeur garde une variable, `cwnd`, la fenêtre de congestion, qui indique combien de données il peut

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc2581.txt`

envoyer sans recevoir d'accusé de réception (ACK). TCP commence avec une fenêtre assez petite (la section 3.1 donne la méthode de calcul, un exposé complet figure dans le RFC 3390) et l'ouvre au fur et à mesure de l'arrivée des accusés de réception (là encore, les valeurs précises figurent dans le RFC).

Si vous lisez le code source du noyau Linux 2.6, ces variables supplémentaires sont déclarées dans include/linux/tcp.h. Tous les noms de fichiers donnés plus loin concernent ce noyau.

Quant la fenêtre de congestion a été augmentée au delà d'un certain seuil, noté ssthresh, le second algorithme, l'Évitement de congestion prend le relais. Cet algorithme augmente différemment, de manière plus rapide, la fenêtre de congestion (quand tout va bien) et réagit à la congestion (détectée par l'expiration d'un délai de garde) en réduisant ssthresh, voire la fenêtre de congestion (ce qui peut la faire tomber en dessous de ssthresh, réactivant le démarrage en douceur). Ce second algorithme est décrit en détail dans la même section 3.1. Pour la mise en œuvre sur Linux, voir le très joli net/ipv4/tcp_cong.c.

Attention, comme l'ont montré plusieurs alertes de sécurité <http://cseweb.ucsd.edu/~savage/papers/CCR99.pdf> (la dernière datant de quelques jours après la publication du RFC <http://www.cisco.com/warp/public/707/cisco-sa-20090908-tcp24.shtml>), le fait de réagir aux messages de l'hôte distant présente toujours un risque, si celui-ci essaie de vous tromper. Diverses contre-mesures sont citées (cf. RFC 3465).

Les deux autres algorithmes, la Retransmission rapide et la Récupération rapide, sont décrits dans la section 3.2. Tous les deux reposent sur les accusés de réception dupliqués. Ceux-ci (définis formellement en section 2) sont des accusés de réception pour des données qui ont déjà fait l'objet d'un tel accusé. Notre RFC 5681 recommande que le TCP récepteur envoie un accusé de réception immédiatement lorsqu'il reçoit des données qui ne sont pas dans l'ordre des numéros de séquence (fonction __tcp_ack_snd_check dans net/ipv4/tcp_input.c). Cet accusé, considéré comme « dupliqué », permet à l'envoyeur de voir tout de suite que des données ont probablement été perdues (rappelez-vous que les accusés de réception, pour TCP, indiquent simplement le rang du dernier octet reçu avec succès, pas une plage ; les données hors-séquence ne suscitent normalement pas d'accusé de réception immédiat, TCP attend plutôt que les données manquantes arrivent).

La Retransmission rapide est utilisée lorsque TCP reçoit des accusés de réception dupliqués. Dans ce cas, TCP doit retransmettre tout de suite les données non encore acceptées par son pair, sans attendre que le délai de garde expire. La Récupération rapide consiste à ne pas utiliser le Démarrage en douceur lorsqu'on détecte des accusés de réception dupliqués. Elle a fait depuis l'objet de perfectionnements, décrits plus longuement dans le RFC 6582.

Après ces quatre algorithmes à utiliser, le RFC, dans sa section 4.1, couvre aussi le problème du redémarrage des sessions qui ont été longtemps inactives. Dans ce cas, TCP ne peut plus compter sur l'arrivée des accusés de réception pour calibrer le réseau (tous les ACK sont arrivés depuis longtemps) et son calcul de la fenêtre de congestion peut être désormais dépassé. Il risque donc d'envoyer un brusque flux de données. Le RFC recommande donc une variante du Démarrage en douceur, lorsque la session a été inactive pendant longtemps.

La section 4.2 revient sur le problème de l'envoi des accusés de réception. Le RFC 1122 (dans sa section 4.2.3.2) spécifie un algorithme pour cet envoi, dont le principe de base est d'attendre un peu avant d'envoyer l'ACK (dans l'espoir que d'autres données arriveront, permettant de se contenter d'un seul ACK), mais sans attendre trop pour éviter que l'émetteur ne s'impatiente et ne réemette. Notre RFC 5681 précise simplement les règles (ambigües) du RFC 1122.

En prime de la Retransmission rapide et du Redémarrage rapide, certains algorithmes ont été proposés et parfois déployés, pour récupérer des pertes de données. La section 4.3 les énumère, sans forcément prendre position sur leur opportunité. Certains, comme celui du RFC 3517, dépendent de l'option `SACK` (”*Selective Acknowledgment*”) du RFC 2018, d'autres, comme celui du RFC 3782 n'en dépendent pas.

Il est important de noter que le bon fonctionnement de l'Internet dépend de la mise en œuvre correcte de tous ces principes, par chaque implémentation de TCP. Deux machines « inciviles » qui désireraient envoyer le plus de données possibles, et tant pis pour la congestion, peuvent techniquement le faire (section 5, consacrée à la sécurité). C'est un des aspects les plus étonnants de l'Internet que peu de vendeurs aient eu l'idée de promouvoir un TCP « optimisé », c'est-à-dire égoïste. On trouve toutefois des cas limites comme Fast TCP.

Enfin, les changements par rapport au RFC 2581 sont présentés dans la section 7. Les principaux sont :

— Une définition rigoureuse de la notion d'accusé de réception dupliqué (en section 2),
— La recommandation d'utiliser le RFC 3465 pendant le Démarrage en douceur,
— La recommandation d'utiliser le RFC 3042 pendant les Retransmission rapide et Récupération rapide.

(Je trouve personnellement que le RFC 2001 était le plus clair de tous et que la précision des mises à jour suivantes a plutôt brouillé les explications.)